GLOBAL EQUATION SOLVER AND OPTIMIZER

Background

[0001] Planning and scheduling require solving and optimizing a set of nonlinear equations. Consider a plant that makes ice cream. The plant needs to make vanilla, strawberry, and chocolate ice cream, but there are a number of constraints, such as the amount of vanilla flavoring, strawberries, and chocolate available. There are scheduled deliveries of ice cream and orders to fill. When a truck rolls up and expects to be loaded with chocolate ice cream, for example. There are production constraints, such as needing to do chocolate last because after producing chocolate ice cream, the pipes must be cleaned before you can run anything else. This is a cost penalty for chocolate. While the plant is making one kind of ice cream it cannot produce anything else, because the equipment is committed to a particular product. Business managers would like to make as large a batch as possible because that minimizes the amount of time the system is down for transitions between products. On the other hand, storage space is limited. For example, chocolate syrup may start to build up. Trucks full of chocolate syrup sit in the plant parking lot and charge thousands of dollars a day until you can get them unloaded.

[0002] In managing plant operations, there are a series of continuous and discreet decisions to make. To get rid of the chocolate syrup, chocolate will need to be produced on Tuesday and strawberry on Wednesday and some time will need to be spent cleaning up after the chocolate. Those are discreet decisions, because a plant cannot only half make chocolate ice cream; the plant either makes it or not. Given a particular set of discrete decisions, a set of continuous decisions is made. How much chocolate ice cream should be produced? How much strawberry ice cream should be produced? How much chocolate is the plant going to have in the tanks at a particular time of day? How much chocolate will be produced in the next hour? Is it going to build up to the point where is

blows the top off the tank? How much chocolate syrup is required? When does the plant start making strawberry? How fast will the strawberries be used up and when will the strawberries run out? Those are continuous decisions. Complications may arise, such when the guy who makes strawberry is late or when it takes longer than usual to clean up after the chocolate. Discrete and continuous decisions like these are used to set up a set of equations or constraints.

[0003] In addition, there are other motivations, such as market demand. A customer may be willing to pay twice as much for all chocolate ice cream as for half chocolate and half strawberry. Perhaps a plant manager is motivated to do a lot more vanilla, even though the supply schedule tells him he is not able to get to vanilla for a long time. Then, he will want to optimize production of vanilla as soon as the vanilla supplies arrive. Even more complex situations occur.

[0004] For a planning system, a set of equations derived from high-level constraints is solved to find the best solution based on certain criteria, such as profitability. This result tells whether the plan will work. If so, then more detailed constraints and criteria are added to the set of equations and the set of equations is solved for a schedule. Scheduling systems have a much larger set of equations to solve than planning systems. This puts a lot more stress on the solver and the solving technology.

[0005] Most current systems are planning systems, not scheduling systems, which are larger and harder to solve. Current methods are unable to solve scheduling problems within the necessary time for rapid modeling and simulation with optimum or near optimum solutions. Also, current methods suffer from a local optimality problem, where a local optimum in the solution space is incorrectly given as the solution when the actual global optimum lies elsewhere. Any system that suffers from the local optimality problem may not be able to find a solution, when a solution exists. In addition, some of the current methods are too slow. Furthermore, most current methods fail to find a

solution without indicating if the problem is infeasible or not. The few methods that are able determine infeasibility have poor or slow convergence. Some examples of current methods are sequential linear programming, sequential quadratic programming, and trusted regions. All three of these suffer from the local optimality problem and when they fail, it is unknown whether the problem is infeasible or not. There is a need for an efficient method to quickly converge on a solution, to rigorously determine whether or not the problem is infeasible, and to find the global optimum instead of getting stuck in a local optimum.

Summary

[0006] The present invention solves both planning and scheduling problems by combining a number of technologies. It balances the safety of subdivision methods with the fast convergence of linearization methods, avoiding the local optimality problem. Also, the linearization methods rigorously determine whether or not the problem was infeasible. The present invention is a method of solving an operations problem. Operations problems comprehend both planning and scheduling problems. First, the method receives variables, relationships, and constraints. Then, it forms a set of non-convex quadratic equations based on them. The method solves the set of non-convex quadratic equations by applying a bound propagation process, a local linear bounding process, a local linearization process, and a global subdivision search. Finally, the method determines whether a solution is optimal, feasible, or infeasible. Thus, the present invention recognizes local optimality problems and goes beyond them to find a global solution, if one exists. If not, the present invention rigorously proves infeasibility. If there is a solution, it comes up with a solution.

Brief Description of the Drawings

[0007] Figure 1 is a block diagram of example applications of the present invention.

Figure 2 is a block diagram of a bounded region containing a solution space to solve for an optima using embodiments of the present invention.

Figure 3 is a flow chart of a method embodiment of the present invention.

Figure 4 is a more detailed flow chart than Figure 3 and shows a method embodiment of the present invention.

Figure 5 is a block diagram of a bounded region, like the one shown in Figure 2, which is modified in a bound propagation and refinement subprocess of a method embodiment of the present invention, such as the method embodiments shown in Figures 3 and 4.

Figure 6 is a flow chart of a local linear bounding subprocess of a method embodiment of the present invention, such as the method embodiments shown in Figures 3 and 4.

Figure 7 is a flow chart of a linearization subprocess of a method embodiment of the present invention, such as the method embodiments shown in Figures 3 and 4.

Detailed Description

[0008] Methods for a global equation solver and optimizer are described. In the following detailed description, reference is made to the accompanying drawings, which form a part hereof. These drawings show, by way of illustration, specific embodiments in which the invention may be practiced. In the drawings, like numerals describe substantially similar components throughout the several views. These embodiments are described in sufficient detail to enable those skilled in the art to practice the invention. Other embodiments may be utilized and structural, logical, and electrical changes may be made without departing from the scope of the present invention.

[0009] Figure 1 is a block diagram of example applications of the present invention. The present invention is shown as a solver 100. The solver solves a set of equations, giving a set of variables and what they are equal to in the

solution, if one exists. These variables are then translated into a business use, such as an operations plan, an accounting summary or some other useful, concrete, and tangible results. The solver is like an octopus, because customer orders interact with inventory, which in turn interacts with production facilities, which interacts with the business plan and so on. The solver is at the heart of all these activities.

[0010] The solver is run as a scheduling system at least on a daily basis. As a planning system, it is run about once a month with adjustments as events occur. Or run to interact with traders as they phone in opportunities. The solver 100 can be applied to scheduling 102, planning 104, and other operations 106.

[0011] Planning is the forecasting of the average performance of a plant (a collection of interconnected processes) over some specified period, such as a month or a year. The plan specifies what inputs are needed and how they are to be used to produce plant outputs. The plan usually includes forecasts of the values of individual process performance parameters such as yields, product qualities, flow rates, temperatures, and pressures. The plan also includes economic information about the impact of changes in parameters on plant profitability.

[0012] Scheduling is the specification of the inputs to and outputs from each process and inventory, plus the timing and sequencing of each production operation, whether batch or continuous, over some short scheduling period, such as a week or 10 days. Although the horizon is a week or 10 days, today's operation is the most important. Operations are not averaged over the scheduling period; rather, time and operations move continuously from the beginning of the period to the end. Ideally, the schedule is revised each day as needed so that it always starts from current actual performance.

[0013] Additionally, the solver can be applied to inventory 108, suppliers 110, and ordering 112. Also, it can be applied to customers 114 and production 116. For example, a business analyst takes a phone call from a trader who says he has

a chance to sell this much of something and asks the business analyst if he can make it in time. The business analyst quickly runs the solver to decide whether it is a good opportunity or not. Another example is storing routine results in an executive information system for general business planning, such as forecasting whether profit numbers will be made for the quarter. Another example is an operations person who uses the results to decide what temperature to crank his unit. Another example is a factory floor scheduler with a cell phone and spreadsheet, who directs operations people based on the results. In addition, the present invention has many other applications.

[0014] Figure 2 is a block diagram of a bounded region 200 containing a solution space 202 to solve for an optima 204 using embodiments of the present invention. Initially, the solution space is unknown, but the bounded region 200 can be determined from the set of equations to solve. The ranges of variables in the set of equations define certain bounds. The exemplary bounded region 200 is shown in three-dimensional space with x- 206, y- 208, and z-axes 210. This represents a rather simple set of equations with 3 variables. A typical set of equations has 10,000 variables, so a typical bounded region would have 10,000 dimensions. However, the three dimensions shown in Figure 2 are easier to conceptualize. In Figure 2, the x variable has lower and upper bounds, LB(X) and UB(X) respectively. The y variable has lower and upper bounds, LB(Y) and UB(Y) respectively. The z variable has lower and upper bounds, LB(Z) and UB(Z) respectively. The bounded region 200 is the intersection of these lower and upper bounds and the solution space 202 is contained within the bounded region 200.

[0015] Assuming the problem is not infeasible, the solution space 202 is a set of feasible solutions in a feasible region, one of which is the optima 204. A feasible region is the set of all points satisfying all of the problem's constraints and restrictions defined in the set of equations. For example, in a feasible solution for an oil refinery, no tanks overflow or underflow. Often, for a

scheduling problem, the goal is to meet the schedule with a feasible solution, rather than an optimal solution. For example, given a certain amount of inventory and shipments, a feasible solution makes the shipments without using up too much of the inventory. Because the deals have already been made, any extra product is stored, not sold. An infeasible problem occurs when the feasible region is empty, i.e. it contains no points. By definition, an infeasible problem has no optimal solution. An optima or optimal solution is a point in the feasible region with the largest objective function value, for a maximization problem. Similarly, for a minimization problem, the optimal solution is a point in the feasible region with the smallest objective function value. An objective function is some function of particular decision variables to be minimized or maximized. For example, a simple objective function to maximize is (weekly revenues) – (raw material purchase costs) – (other variable costs). Some other examples of optimality measures or objective functions are minimum turnover, minimum costs, accomplishing work in minimum time, maximizing a high margin product, and the like. The optima is defined by pre-determined criteria, such as least cost or most profit that are included as input to the present invention. A global optima is the optima over the entire range of variables as opposed to a local optima, which is the optima only in a local area.

[0016] Figure 3 is a flow chart of a method embodiment 300 of the present invention. One aspect of the present invention is a method 300 of solving an operations problem. Operations problems comprehend both planning and scheduling problems. For example, problems include maximizing profits, meeting shipments, meeting production schedules, meeting product specification requirements, and other business problems. Operations problems are optimization problems in industries as diverse as banking, education, forestry, petroleum, and trucking. The method 300 comprises receiving variables 302, relationships 304, and constraints 306. The variables 302 are things like qualities, quantities, timing, and the like. Relationships 304 express how the

variables 302 interrelate, such as how speed relates to quality. For example, in mixing things, the relative quantities effect the quality of the mixture. Other examples are relationships 304 based on the physics of a plant and relationships 304 based on economics, such as cost and revenue. Some examples of constraints 306 in refinery applications are tank limits, product specifications, gasoline octane ratings, operating limits and the like.

[0017] The method 300 comprises forming a set of non-convex quadratic equations 308 based on the variables 302, relationships 304, and constraints 306. Some example equations are formed by the definition of the problem. Other example equations are formed by physical constraints, such as the capacity of a tank. Still other example equations are formed by the physics of a process being modeled. Equations that are not in quadratic form can be converted to quadratic form by standard approximation methods. Convex equations have solution sets such that a line segment joining any pair of points in the solution set is wholly contained in the solution set. Non-convex equations do not have this property. Generally, when a convex object, such as a beach ball is put on a flat surface, it will be touching at one point. Non-convex objects basically have some indentation or dimple in it. Non-convex equations usually have multiple local pseudo-solutions, meaning they are not solvable with current methods having the local optimality problem.

[0018] The method 300 further comprises solving the set of non-convex quadratic equations by applying a bound propagation process, a local linear bounding process, a local linearization process, and a global subdivision search. This is shown in Figure 3 as the solver 310. Solving the equations is described in more detail below with reference to Figure 4. The method 300 further comprises determining whether a solution is optimal, feasible, or infeasible. Figure 3 shows the solver 310 determines whether a solution exits 312. If no solution exists, then the solver 310 determines that the problem is infeasible 314. If a solution exists, then the solver 310 determines whether the solution is

optimal 316. The solver determines either that the solution is optimal 318 or feasible 320.

[0019] The solution to the set of non-convex quadratic equations has many uses. (See Figure 1.) In one embodiment, the solution is a schedule for the manufacturing process. In another embodiment, the solution is a schedule for operating an oil refinery. In another embodiment, the solution is a plan for the manufacturing process. In another embodiment, the solution is a plan for operating an oil refinery.

[0020] Figure 4 is a more detailed flow chart than Figure 3 and shows a method embodiment 400 of the present invention. One aspect of the present invention is a method 400 comprising solving the set of non-convex quadratic equations by applying a bound propagation process 402, a local linear bounding process 404, a local linearization process 406, and a global subdivision search 408. Three of these processes are described below with reference to Figures 5-7. The bound propagation process 402 is described below with reference to Figure 5. The local linear bounding process 404 is described below with reference to Figure 6. The local linearization process 406 is described below with reference to Figure 7.

[0021] The global subdivision search 408 is done over a bounded region, like the one shown in Figure 2 using branch and prune logic. For branching, the solver starts with the whole bounded region and if no solution is found, splits the region in half, first searching one half and then the other until a solution is found or the problem is determined to be infeasible. For pruning, only bounded regions that might contain a solution need to be examined. There are many alternate methods for performing the global subdivision search. One alternate method is to pick the most infeasible constraint and find the most infeasible variable in it.

[0022] The combination of processes in method 400 work together to create advantages over current methods. The bound propagation process 402 gives the method 400 improved performance over current methods. The local linear bounding process 404 rigorously proves infeasibility locally. The local

linearization process 406 improves time to convergence on a solution over current methods. The global subdivision search 408 helps to avoid the local optimality problem. These advantages help reduce project cycles, integrate operations for better decisions, and increase profits by providing more accurate, timely information for real-time decisions.

[0023] Another aspect of the present invention is a machine-accessible medium having associated content capable of directing the machine to perform a method 400 of solving a set of non-convex quadratic equations. At the start 410 in Figure 4, the method 400 comprises selecting a region bounding all variables 412. A bound propagation process is applied to the region to refine the bounds and improve linearization 402. A local linear bounding process is applied to the region to determine feasibility and to find approximately feasible solutions 404. The local linear bounding process is optionally repeated. The method 400 determines if there is no potential for a solution 414 within the region. If so, the region is eliminated from consideration 416. The method 400 determines if there is no potential for an optimal solution 418. If so, the region is eliminated from consideration 420.

[0024] A local linearization process is applied to the region to determine feasibility and local optimality 406. The local linearization process is optionally repeated. The method 400 determines if there is a solution 422 and if so, if it is optimal 424. Upon finding an optimal global solution 426, the optimal global solution and information indicating optimality are provided and the method 400 ends 428. Upon finding a feasible global solution 430, the feasible global solution and information indicating feasibility are provided and the method 400 ends 428. Upon determining local infeasibility, the region is eliminated from consideration. Upon determining global infeasibility 432, information indicating infeasibility is provided and the method 400 ends 428. Upon not finding a solution, a global subdivision search 408 is applied to the region to produce two or more regions and the bound propagation 402, local linear bounding 404, and

local linearization 406 processes are iteratively applied to each of the two or more regions, until the solution is determined to be optimal 426, feasible 430, or infeasible 432.

[0025] In one embodiment, the method 400 further comprises receiving input variables, constraints, and equations. In another embodiment, the method 400 further comprises receiving a measure of optimality used to determine the global optimal solution. In another embodiment, the method 400 further comprises receiving a measure of feasibility used to determine the global feasible solution. In another embodiment, the method 400 further comprises providing a schedule for operating a plant. In another embodiment, the method 400 further comprises providing a plan for operating a plant.

[0026] Another aspect of the present invention is a process 400 of solving a set of non-convex quadratic equations. The process 400 comprises selecting a region bounding all variables 412. A bound propagation process 402 is applied to the region to refine the bounds and improve linearization. A local linear bounding process 404 is applied to the region to determine feasibility and to find approximately feasible solutions. A local linearization process 406 is applied to the region to determine feasibility and local optimality. Upon finding a solution after the local linearization process, the solution is provided. Upon determining infeasibility, the region is eliminated from consideration. Upon not finding the solution after the local linearization process, a global subdivision search 408 is applied to the region to produce two or more regions and the bound propagation 402, local linear bounding 404, and local linearization 406 processes are iteratively applied to each of the two or more regions, until it is determined that the solution is optimal, feasible, or infeasible.

[0027] In one embodiment of the process 400, the local linearization process 406 is the local linear bounding process 404.

[0028] Figure 5 is a block diagram of a bounded region, like the one shown in Figure 2, which is modified in a bound propagation and refinement subprocess of

a method embodiment of the present invention, such as the method embodiments shown in Figures 3 and 4. Figure 5 shows a bounded region 500 with refined upper and lower bounds along the y-axis to produce a refined region 502, which is closer to the solution space 504 and reduces the space to search. Once bounds are refined, they are propagated. For example, if one of the equations is X = Z + Y, once Y is bound, then that can be propagated to X and Z so that X and Z are also bound, reducing the size of the region to search.

[0029] Figure 6 is a flow chart of a local linear bounding subprocess 600 of a method embodiment of the present invention, such as the method embodiments 300, 400 shown in Figures 3 and 4. In one embodiment, the local linear bounding subprocess 600 determines infeasibility and finds approximately feasible solutions. In one embodiment, the local linear bounding process 600 comprises performing differentiation 602 on equations 604 in the region 606. The region 606 includes an initial point (x_0). Differentiation 602 is done around the x_0. The lower and upper bounds on the variables in the region are determined 608. A linear programming process is applied to the linear equations in the region 610. The local linear bounding subprocess 600 determines whether a solution exists in the region 612. Upon finding a solution exists, local feasibility is determined 614. Upon finding local infeasibility, global infeasibility is determined 616.

[0030] In one embodiment, the initial point is a trial solution closest to a feasible solution. In another embodiment, the initial point is a trial solution closest to being the optimal solution. In another embodiment, the initial point is the largest. In another embodiment, the initial point is the smallest. Other embodiments include using discrete search techniques.

[0031] Figure 7 is a flow chart of a linearization subprocess 700 of a method embodiment of the present invention, such as the method embodiments 300, 400 shown in Figures 3 and 4. In one embodiment, the local linearization process 700 comprises performing differentiation 702 at a point in the bounded region

704. A set of linear equations is formed 706. A linear programming process is applied to the linear equations in the bounded region 708. A new point is generated in the bounded region 710 and the local linearization process is repeated with the new point. If the linear program fails to generate a new point 712, then the global subdivision search or some other method is applied 714. In another embodiment, the local linearization process 700 has quadratic convergence, giving it fast performance.

Example Embodiments

[0032] The following example embodiments provide methods for the global solution and optimization of systems of quadratic equations, with extensions to general nonlinear functions.

Basic Definitions

[0033] Let $f(x):\Re^n \to \Re^m$ be a quadratic function of the form

$$(1.1) \quad f_k(x) = C_k + \sum_i A_{ki} x_i + \sum_{ij} B_{kji} x_j x_i \,.$$

The problem we wish to solve will have the following form.

$$(1.2) \quad \begin{aligned} &\min f_k(x) : k \in K_0 \\ &lb_k \le f_k(x) \le ub_k : \forall k \in K_1 \\ &u^0 \le x \le v^0 \end{aligned}$$

(The set K_0 must include only a single linear function.)

[0034] With informal notation, we shall write the functions in the matrix/vector notation

$$(1.3) \quad f(x) = C + Ax + Bxx$$

when we wish to consider all the functions, and in the forms

$$(1.4) \quad \begin{aligned} f^0(x) &= C^0 + A^0 x \\ f^1(x) &= C^1 + A^1 x + B^1 xx \end{aligned}$$

when we wish to consider the three classes of functions.

[0035] Notationally, Bxy is to be interpreted as a multilinear

form $Bxy = \sum_{ij} B_{kji} x_j y_i$ and the linear form Bx is equivalent to the matrix $(Bx)_{ki} = \sum_j B_{kji} x_j$. In addition, the transpose of B is defined as $B^* xy = Byx$, with the immediate equality $B^*_{kji} = B_{kij}$.

[0036] Define the positive and negative functions

$$(1.5) \quad \begin{aligned} (x)_+ &= \max(x,0) \quad \geq 0 \\ (x)_- &= \min(x,0) \quad \leq 0 \end{aligned}$$

[0037] We define a measure of the *infeasibility* of a particular point to be

$$(1.6) \quad \Delta(x) = \sum_{k \in K_1} \Delta_k(x)$$

where

$$(1.7) \quad \Delta_k(x) = \max\left((f_k(x) - ub_k)_+, (lb_k - f_k(x))_+\right).$$

(As a result, $\Delta(x) \geq 0$, and $\Delta(x) = 0$ only if the point x is feasible.)

[0038] In the course of solving the problem (1.2) above, we will be solving a sequence of subsidiary problems. These problems will be parameterized by a trial solution and a set of point bounds,

$$(1.7) \quad \begin{matrix} \bar{x} \\ u \leq x \leq v \end{matrix}.$$

From that we can compute a set of gradient bounds

$$(1.8) \quad \begin{aligned} F &= (B)_+ u + (B)_- v \\ G &= (B)_+ v + (B)_- u \end{aligned}$$

so that

$$(1.9) \quad u \leq x \leq v \;\Rightarrow\; F \leq Bx \leq G.$$

We will also require that the trial solution also satisfy the bounds

$$(1.10) \quad u \leq \bar{x} \leq v.$$

[0039] We define the row vector $e = \{1,...,1\}$ so that we can compute the *maximum infeasibility* for the bounds

$$(1.11) \quad \Delta(u,v) = e(G-F)(v-u) = \sum_{ki} (G_{ki} - F_{ki})(v_i - u_i)$$

or equivalently

$$(1.12)\quad \Delta(u,v) = e|B|(v-u)(v-u) = \sum_{kji} |B_{kji}|(v_j - u_j)(v_i - u_i).$$

[0040] In the course of the method, we will also be computing a series of trial solutions. We can compute the *divergence* of a sequence of trial solutions $\{\bar{x}^1, \bar{x}^2, \ldots\}$

$$(1.13)\quad \left\|\bar{x}^2 - \bar{x}^1\right\| = \sum_i \left|\bar{x}_i^2 - \bar{x}_i^1\right|.$$

The centered representation of a function relative to a given trial solution $\bar{x}$ is

$$(1.14)\quad f(x) = \overline{C} + \overline{A}(x-\bar{x}) + B(x-\bar{x})(x-\bar{x})$$

where

$$(1.15)\quad \begin{aligned} \overline{A} &= A + (B + B^*)\bar{x} \quad \left(= A_{ki} + \sum_j (B_{kji} + B_{kij})\bar{x}_j\right) \\ \overline{C} &= C + A\bar{x} + B\bar{x}\bar{x} \quad \left(= C_k + \sum_i A_{ki}\bar{x}_i + \sum_{ij} B_{kji}\bar{x}_j\bar{x}_i\right) \end{aligned}.$$

[0041] By also defining

$$(1.16)\quad \begin{aligned} \overline{u} &= u - \bar{x} \\ \overline{v} &= v - \bar{x} \\ \overline{F} &= F - B\bar{x} \\ \overline{G} &= G - B\bar{x} \end{aligned}$$

the bounding inequalities are equivalent to the following centered inequalities

$$(1.17)\quad \begin{aligned} \overline{u} \le x - \bar{x} \le \overline{v} \\ \overline{F} \le B(x-\bar{x}) \le \overline{G} \end{aligned}.$$

[0042] Also note that since $\overline{u} \le 0 \le \overline{v}$, we have

$$(1.18)\quad |x_i - \bar{x}_i| \le \max(|\overline{u}_i|, |\overline{v}|_i) \le v_i - u_i.$$

In order to bound on both side, we will be decomposing $x - \bar{x}$ into two nonnegative variables,

$$(1.18)\quad \begin{aligned} z - w &= x - \bar{x} \\ z &\ge 0 \\ w &\ge 0 \end{aligned}$$

to which we will apply the bounds

$$(1.19)\quad z_i + w_i \le \max(|\overline{u}_i|, |\overline{v}|_i).$$

[0043] The subsidiary problems are then

A) the basic quadratic feasibility problem—

$$(1.20)\quad P0 = \left\{ x: \begin{array}{l} u^0 \le x \le v^0 \\ lb \le f^1(x) \le ub \end{array} \right\}$$

and its optimization form

$$(1.21)\quad PO0 = \left\{ x: \begin{array}{l} \min f^0(x) \\ x \in P0 \end{array} \right\}.$$

B) the basic quadratic problem with the bounding inequalities-

$$(1.23)\quad Pbd(u,v) = \left\{ x: \begin{array}{l} u \le x \le v \\ lb \le f^1(x) \le ub \end{array} \right\}$$

and its optimization form

$$(1.24)\quad PObd(u,v) = \left\{ x: \begin{array}{l} \min f^0(x) \\ x \in Pbd(u,v) \end{array} \right\}.$$

C) the enveloping linear programming problem (using the formulas of (1.8), (1.15), and (1.16))—

$$(1.25)\quad LP(\overline{x},u,v) = \left\{ x,z,w: \begin{array}{l} \overline{u} \le x - \overline{x} \le \overline{v} \\ lb \le \overline{C}^1 + \overline{A}^1(x-\overline{x}) + \overline{G}^1 z - \overline{F}^1 w \\ ub \ge \overline{C}^1 + \overline{A}^1(x-\overline{x}) + \overline{F}^1 z - \overline{G}^1 w \\ x - \overline{x} = z - w \\ z + w \le \max(|\overline{v}|, |\overline{u}|) \\ z, w \ge 0 \end{array} \right\}$$

its optimization form

$$(1.26)\quad LPO(\overline{x},u,v) = \left\{ x: \begin{array}{l} \min f^0(x) = A^0 x \\ x \in LP(\overline{x},u,v) \end{array} \right\}$$

and its minimal infeasibility form

$$(1.27)\quad LPMI(\overline{x},u,v) = \left\{ x: \begin{array}{l} \min e(G-F)(z+w) \\ x \in LP(\overline{x},u,v) \end{array} \right\}.$$

E) and finally the linearization problem with the bounding inequalities included

$$(1.28)\quad LLP(\bar{x},u,v)=\left\{\begin{array}{ll} & \min A^0 x \\ x: & \bar{u}\le x-\bar{x}\le\bar{v} \\ & lb\le \bar{C}^1+\bar{A}^1(x-\bar{x})\le ub \end{array}\right\}.$$

Method

[0044] The problems above have the following relationships.

$Pbd(u,v)\subseteq LP(\bar{x},u,v)$

If $x'\in LP(\bar{x},u,v)$ then

$Pbd(u,v)\cap LP(\bar{x},u,v)=Pbd(u,v)\cap LP(x',u,v)$

If $x'\in LP(\bar{x},u,v)$ then

$\Delta(x')\le e(G-F)(z'+w')\le\Delta(u,v)=e(G-F)(v-u)$

(Note that it is also true that if $G_{ki}-F_{ki}=0$, then $\overline{G}_{ki}=\overline{F}_{ki}=(B\bar{x})_{ki}=0$, and that particular quadratic term will disappear from the constraint approximations in $LP(\bar{x},u,v)$.)

[0045] As we search for a solution for a problem $PO0$, we split the region up into nodes, each of which is given by a set of bounds $\{u,v\}$.

[0046] For any problem, the theory of Newton's method tells us that there is a constant $\theta>0$ such that for any $\Delta(u,v)<\theta$, if $Pbd(u,v)$ is feasible then the sequence of trial solutions generated by the linearization, $\bar{x}^{m+1}=LLP(\bar{x}^m,u,v)$, will converge quadratically to the solution $x^*=Pbd(u,v)$, with

$$\left\|\bar{x}^{m+2}-\bar{x}^{m+1}\right\|\le\alpha\left\|\bar{x}^{m+1}-\bar{x}^m\right\|^2.$$

[0047] On the other hand by the theorems proven here, there is a constant $\theta>0$ such that for any $\Delta(u,v)<\theta$, if $Pbd(u,v)$ is infeasible then $LP(\bar{x},u,v)$ will be infeasible.

[0048] So a simplistic view of the method is to

1) Choose a node $\{u,v\}$, and find a trial solution $u\le\bar{x}\le v$,

2) Run $\bar{x}^{m+1}=LLP(\bar{x}^m,u,v)$ and see if it finds a feasible point,

3) Run $x \in LP(\bar{x},u,v)$ and see if it is infeasible,

4) If the node fails both (2) and (3), subdivide the node into smaller regions, and try again.

Expanding a Node

[0049] An open node specifies a particular trial solution and bounds, $\{\bar{x},u,v\}$. It also specifies a particular distribution of the quadratic terms $B+B^*$ (see the Symmetry Breaking of the Gradient Terms section below).

[0050] To expand the node, apply the procedure below until the node is fully expanded, or has been closed. The procedure will update the problem, its trial solution and bounds, will establish the status of the node, and will update a rigorous lower bound $\bar{\varphi}$ of the objective function on the node.

[0051] Note that in the course of searching for a suitable node to expand (see section below), it may be desirable to apply only steps (1), (2), (3), and (4) in order to see if there exists a linearization node which succeeds in step (2). This will be referred to as *partial expansion*.

[0052] There are two basic patterns of application of these steps—

A) Run linearization alone as long as possible—

Run steps (1), (2), (3), and (4) when partially expanding a node. Run steps (5) and (6) when fully expanding a node.

B) Always seed linearization with a minimal infeasibility trial solution-

Run steps (1), (4), (5), (2) and (3) when partially expanding a node. Run steps (5) (rerun) and (6) when fully expanding a node. Step (2) should include multiple iterations of linearization, otherwise the quadratic convergence property will be lost.

[0053] Let $\varepsilon > 0$ be the basic numerical tolerance desired.

[0054] 1) Propagate the bounds through the problem (see the Convergence and Divergence of Trial Solutions section below). Update the node with the new bounds and/or problem terms—

$$\{u,v\} = \{u',v'\}$$
$$\{C,A,B\} = \{C',A',B'\}$$

[0055] 2) Try to solve the linearization problem $LLP(\bar{x},u,v)$.

Optionally, one can solve a series of linearization problems, $\bar{x}^{m+1} = LLP(\bar{x}^m,u,v)$. The series should be ended when:

The problem becomes infeasible;

A fixed upper limit of iterations is reached;

The trial solutions converge; or

The trial solutions diverge.

At the end of this series, if there exists a solution $x' \in LLP(\bar{x},u,v)$, then use the solution as the new trial solution $\bar{x} = x'$, and declare the node *linearized.*

[0056] If there were a series of trial solutions found, then if the solutions converges or the upper limit reached, choose the final trial solution, but if the problem became infeasible or the trial solutions diverged, choose the trial solution with the minimum infeasibility, $\bar{x}' = \arg\min_m \Delta(\bar{x}^m)$.

[0057] If the series was ended because the solutions converged, declare the node *convergent.*

[0058] If the series was ended because the solutions diverged, declare the node *divergent.*

[0059] In any case, set the optimal lower bound to the worst bound on the node, $\bar{\varphi} = (A^0)_+ u + (A^0)_- v$. (This will be reset in step 6, but is set here to support partial expansion.)

[0060] 3) If step (2) succeeds in determining a trial solution, compute the infeasibility of the new trial solution, $\Delta(\bar{x})$, and the maximum infeasibility of the bounds $\Delta(u,v)$.

[0061] If $\Delta(u,v) \le \varepsilon$, all points satisfying $x' \in Bd(\bar{x},u,v)$ are feasible within the desired tolerance. Declare the node *completely feasible* and *point-optimal.*

If $\Delta(\bar{x}) \le \varepsilon$, then the trial solution is feasible within the desired tolerance.

Declare the node *point-feasible*. (Note that completely feasible implies point-feasible.) (Note: if the node was convergent, it should be point-feasible.)
If feasibility is all that is desired (as opposed to optimality), make the following substitution for steps (4)-(6)— If the node has been declared point-feasible, then close the node. If not, the node is completely expanded. Exit this procedure in either case.

[0062] If the node is linearized and point-feasible, then by the standard theory of Newton iteration, the trial solution is an approximate local solution to the nonlinear bounded problem $Bd(u,v,F,G)$. Close the node and declare it *point-optimal*, and set the optimal lower bound for the node to the linearized optimum, $\overline{\varphi} = A^0\overline{x}$.

[0063] Alternately, in the case where there are multiple local solutions within the same bounds, determine a local region within which the point is optimal, and close that newfound node, declaring it as point-optimal and point-feasible. Add a constraint to the original node such that the optimum must be strictly better than the found local optimum, $A^0 x < \overline{\varphi} - \varepsilon$ and treat it as a new unopened node.

[0064] 4) If step (2) fails, try to solve the enveloping minimal infeasibility problem $LPMI(\overline{x},u,v)$.

[0065] If no such solution exists, close the node and declare it *infeasible*. Exit this procedure.

[0066] If there exists a solution $x' \in LPMI(\overline{x},u,v)$, then use the solution as the new trial solution $\overline{x} = x'$. (Also record the auxiliary variables $\{z', w'\}$.)

[0067] 5) Compute the infeasibility of the new trial solution, $\Delta(\overline{x})$, and the maximum infeasibility of the bounds $\Delta(u,v)$.

[0068] If $\Delta(u,v) \le \varepsilon$, all points satisfying $x' \in LPMI(\overline{x},u,v)$ are feasible within the desired tolerance. Declare the node *completely feasible*.

[0069] If $\Delta(\overline{x}) \le \varepsilon$, then the trial solution is feasible within the desired tolerance. Declare the node *point-feasible*. (Note that completely feasible implies point-

feasible.)

[0070] 6) If the node is not linearized, and is not infeasible, solve the enveloping optimal problem, $x'' \in LPO(\bar{x},u,v)$. Set the optimal lower bound to the resulting minimum, $\bar{\varphi} = A^0 x''$.

[0071] If the node is completely feasible, then use the new solution as the new trial solution $\bar{x} = x''$, close the node, and declare it *point-optimal.* Exit this procedure.

[0072] If the node is point-feasible and the lower bound is close enough to the trial solution, $\left|\bar{\varphi} - A^0 \bar{x}\right| \le \varepsilon$, close the node, and declare it *point-optimal.* (Do NOT use the new solution as the trial solution.) Exit this procedure.

Otherwise, the node is completely expanded. Exit this procedure.

[0073] Alternately, if feasibility is all that is desired (as opposed to optimality), make the following substitution for step (6)— If the node has been declared point-feasible, then close the node. If not, the node is completely expanded. Exit this procedure in either case.

Choosing, Expanding, and Splitting a Node

[0074] Define φ^* be the current best optimum found at a particular point in the search. That is

$$\varphi^* = \begin{cases} \min(\bar{\varphi}(N) : N \text{ is a point - optimal node}) \\ \infty \text{ if no point - optimal node exists} \end{cases}$$

Let $\varepsilon > 0$ be the basic numerical tolerance desired. Then we proceed as follows with the list of non-closed nodes.

[0075] 1) For all non-closed nodes N, if $\bar{\varphi}(N) \ge \varphi^* - \varepsilon$, close the node and declare it *sub-optimal.* Also, we can close those with bounds equal within tolerance to the best as well as those with bounds strictly greater. Alternately, if feasibility is all that is desired (as opposed to optimality), this step can be skipped.

[0076] 2) Choose a candidate set of nodes from the set of non-closed nodes according to the following.

[0077] If there are linearized nodes, select that set and proceed to step (3).

[0078] If there are nodes that have not yet been partially expanded, then partially expand nodes until either one is linearized, or all have been partially expanded, then go back to step (1).

[0079] At this point, one may optionally choose to fully expand all non-closed nodes, if there are any that need it, and go back to step (1).

[0080] If there are no fully expanded non-closed nodes and there are any non-closed nodes that have not been fully expanded, expand at least one of them and go back to step (1).

[0081] Select the set of expanded nodes and proceed to step (3).

[0082] Else all nodes have been closed, and you may terminate the procedure.

[0083] 3) From the set of non-closed nodes, we wish to select one to split. From the set of chosen nodes, select an individual node according to one of the following possible measures.

The node with the smallest infeasibility of the trial solution, $\Delta(\bar{x})$.

The node with the smallest $\bar{\varphi}$.

The node with the largest maximum infeasibility of the bounds, $\Delta(u,v)$.

These are suggested heuristics. Correctness only requires the expansion of an open node.

[0084] 4) It is now time to subdivide the node.

[0085] We will subdivide the point range. Compute $w = \frac{(u+v)}{2}$.

Optionally, one could also choose $w = \bar{x}$ if one wants to subdivide at the points of linearization.

[0086] If the node is linearized, and divergent, we will compute the worst divergence

$$i^* = \arg\max_{\{i\}} \left| \bar{x}_i^{m+1} - \bar{x}_i^m \right| .$$

If the node is not linearized, , we will compute the dimension of the largest infeasibility according to the trial solution.

$$i^* = \arg\max_{\{i\}} \sum_k (G_{ki} - F_{ki})(z_i + w_i) .$$

Otherwise, we can compute the dimension of the largest infeasibility according to the point bounds.

$$i^* = \arg\max_{\{i\}} \sum_k (G_{ki} - F_{ki})(v_i - u_i) .$$

[0087] 5) Close the node, and open two new nodes with the same problem, trial solution, and bounds are the newly closed node, except that for the first new open node,

$$u_i' = u_i : \forall i$$
$$v_i' = v_i : \forall i \neq i^*$$
$$v_{i^*}' = w_{i^*} + \varepsilon/10$$

and for the second new open node,

$$u_i' = u_i : \forall i \neq i^*$$
$$u_{i^*}' = w_{i^*} - \varepsilon/10 \quad .$$
$$v_i' = v_i : \forall i$$

Initialization and Termination

[0088] To start the list of nodes, we require a problem $\{C, A, B\}$, a realistic set of finite bounds with a reasonable trial solution $-\infty < u \leq \bar{x} \leq v < \infty$ and a basic

error tolerance ε.

[0089] To terminate the procedure, we continue to split, expand, and close nodes until all nodes have been closed. At that point we have either:

1) There exists a point-optimal node whose value is as minimal as any other node, and thus the trial solution of that node is the optimum solution of the original problem; or

2) All nodes are infeasible and so is the original problem.

Alternately, if all we are interested in is feasibility, we only need look for any node that is point-feasible, in which case the trial solution is the feasible point.

Convergence and Divergence of Trial Solutions

[0090] By the theory associated with Newton's method, if the bounded problem is feasible, and the initial trial solution is "sufficiently" close to the solution $x^* = Pbd(u,v)$, then the series of trial solutions will be quadratically convergent, with $\left\|\bar{x}^{m+2} - \bar{x}^{m+1}\right\| \le \alpha \left\|\bar{x}^{m+1} - \bar{x}^{m}\right\|^2$. If the problem is infeasible, or if the initial trial solution is not sufficiently close, then examples from chaos theory show that almost any behavior may occur.

[0091] So our criterion for convergence or divergence will be the following.

The trial solutions diverge if $\left\|\bar{x}^{m+2} - \bar{x}^{m+1}\right\| > \left\|\bar{x}^{m+1} - \bar{x}^{m}\right\|$.

The trial solutions converge if $\left\|\bar{x}^{m+2} - \bar{x}^{m+1}\right\| \le \varepsilon$.

Otherwise, we pass no judgment.

The measures used for convergence/divergence are purely pragmatic, and the metric $\|\cdot\|$ can be chosen for convenience, so that for example either the sum of difference, or the maximum difference can be used.

Simple Propagation of Bounds

[0092] Iterate the following propagations until no further improvement is seen. Alternately, include the square-free bound and the non-square-free bound

propagations, until no improvement is found.

[0093] 1) If there exists a $\{k',i'\}:(G_{ki}-F_{ki})<\varepsilon$, then to within the error tolerance, we can assume that

$$(Bx)_{k'i'}=\frac{(F_{k'i'}+G_{k'i'})}{2}$$

and we can then remove a quadratic term from the problem by adding the linear constraint

$$0=-\frac{(F_{k'i'}+G_{k'i'})}{2}+\sum_j B_{k'ji'}x_j$$

and modifying the original constraint

$$f_{k'}(x)=C_{k'}+\sum_i A_{k'i}x_i+\sum_{ij} B_{k'ji}x_jx_i$$

to

$$f_{k'}(x)=C_{k'}+\sum_i A_{k'i}x_i+\frac{(F_{k'i'}+G_{k'i'})}{2}x_{i'}+\sum_{i\neq i'j} B_{k'ji}x_jx_i\,.$$

[0094] Let the new constraint have new index $\hat{k}$. Then

$$C'_k=C_k:\forall k\neq k'$$

$$C'_{k'}=-\frac{(F_{k'i'}+G_{k'i'})}{2}$$

$$A'_{ka}=A_{ki}:\forall k\neq k'\ \forall i$$

$$A'_{k'i}=A_{k'i}:\forall i\neq i'$$

$$A'_{k'i'}=A_{k'i'}+\frac{(F_{k'i'}+G_{k'i'})}{2}$$

$$A'_{\hat{k}i}=B_{k'ii'}:\forall i$$

$$B'_{kji}=B_{kji}:\forall k\neq k'\ \forall j\ \forall i$$

$$B'_{k'ji}=B_{k'ji}:\forall j\ \forall i\neq i'$$

$$B'_{k'ji'}=0:\forall j$$

$$B'_{\hat{k}ji}=0:\forall j\ \forall i$$

[0095] 2) For every linear constraint

$$k:f_k(x)=C_k+\sum_i A_{ki}x_i$$

we can determine new point bounds

$$\forall k \in K_1, f_k \text{ linear}, \forall i: \begin{cases} A_{ki} > 0 \Rightarrow v_i' = \min\left(v_i', \dfrac{ub_k - C_k' - \sum_{j \neq i} (A_{kj})_+ u_j' - \sum_{j \neq i} (A_{kj})_- v_j'}{A_{ki}}\right) \\ A_{ki} < 0 \Rightarrow v_i' = \min\left(v_i', \dfrac{lb_k - C_k' - \sum_{j \neq i} (A_{kj})_+ v_j' - \sum_{j \neq i} (A_{kj})_- u_j'}{A_{ki}}\right) \end{cases}$$

$$\forall k \in K_1, f_k \text{ linear}, \forall i: \begin{cases} A_{ki} > 0 \Rightarrow u_i' = \max\left(u_i', \dfrac{lb_k - C_k' - \sum_{j \neq i} (A_{kj})_+ v_j' - \sum_{j \neq i} (A_{kj})_- u_j'}{A_{ki}}\right) \\ A_{ki} < 0 \Rightarrow u_i' = \max\left(u_i', \dfrac{ub_k - C_k' - \sum_{j \neq i} (A_{kj})_+ u_j' - \sum_{j \neq i} (A_{kj})_- v_j'}{A_{ki}}\right) \end{cases}$$

Symmetry Breaking of the Gradient Terms

[0096] We can exploit the fact that the original quadratic functions only are affected by the value $B + B^*$ in order to attempt to reduce the degree of nonlinearity. This should be applied sparingly, as every time this rule is applied the gradient bounds deteriorate. It should definitely be applied at the initialization of the method.

[0097] Even if the heuristic below is not applied, the B used should be upper triangular under some permutation of the variables. That is, for any constraint k and variable pair $\{i, j\}$, either $B_{kij} = 0$ or $B_{kji} = 0$. This is to minimize the number of terms actually present in the quadratic expressions.

[0098] To apply the rule in the initial step, we redistribute the symmetric elements in B according to the heuristic rule of putting all the quadratic "weight" on the variable with the smallest range (break ties lexicographically)—

(Initialization at iteration 0.)

$$\forall\{k,i,j\}: \text{if } |v_j - u_j| < |v_i - u_i|, \text{ then } \begin{cases} B'_{kij} = B_{kij} + B_{kji} \\ B'_{kji} = 0 \end{cases}.$$

[0099] To apply the rule in subsequent step, we redistribute the symmetric elements in B only if there is a significant difference between the variables' ranges—

(At iteration m)

$$\forall\{k,i,j\}: \text{if } |v_j - u_j| < 10 * |v_i - u_i| \text{ and } B_{kji} \neq 0, \text{ then } \begin{cases} B'_{kij} = B_{kij} + B_{kji} \\ B'_{kji} = 0 \end{cases}.$$

Square-Free Bound Propagation

[0100] Assume we have a quadratic function inequality

$$lb_k \leq f_k(x) = C_k + \sum_i A_{ki} x_i + \sum_{ij} B_{kji} x_j x_i \leq ub_k$$

a set of point bounds $u \leq x \leq v$, and wish to refine the bounds for a variable x_J. We will ignore any benefits of explicitly considering square terms x_J^2 in the function f_k, (hence "square-free propagation") although we will allow squared terms for variables other than x_J to appear. The Non-Square-Free Bound Propagation section explores the additional benefits of explicitly considering x_J^2.

[0101] In order to apply the Lemmas (see section below), we first rearrange the inequalities into the form

$$lb_k - C_k \leq A_{kJ} x_J + \left(\sum_{i \neq J} (B_{kJi} + B_{kiJ}) x_i + B_{JJ} x_J \right) x_J + \sum_{i \neq J} A_{ki} x_i + \sum_{i \neq J, j \neq J} B_{kji} x_j x_i \leq ub_k - C_k$$

and so if we define

$$\beta = \left(A_{kJ} + \sum_{i \neq J} (B_{kJi} + B_{kiJ}) x_i + B_{JJ} x_J \right)$$

$$\gamma = \beta x_J$$

we also have

$$lb_k - C_k - \sum_{i \neq J} A_{ki} x_i - \sum_{i \neq J, j \neq J} B_{kji} x_j x_i \leq \gamma \leq ub_k - C_k - \sum_{i \neq J} A_{ki} x_i - \sum_{i \neq J, j \neq J} B_{kji} x_j x_i$$

[0102] From the Bounding Lemmas section, we define the bounding functions for multiply and square—

$$\mu xy(x_0, x_1, y_0, y_1) = \min(x_0 y_0, x_1 y_0, x_0 y_1, x_1 y_1)$$

$$vxy(x_0, x_1, y_0, y_1) = \max(x_0 y_0, x_1 y_0, x_0 y_1, x_1 y_1)$$

$$\mu xx(x_0, x_1) = \min(x_0^2, (x_0 x_1)_+, x_1^2)$$

$$vxx(x_0, x_1) = \max(x_0^2, (x_0 x_1)_+, x_1^2).$$

We then define bounds $\beta_0 \leq \beta \leq \beta_1$ and $\gamma_0 \leq \gamma \leq \gamma_1$ for $\{\beta, \gamma\}$—

$$\beta_0 = A_{kJ} + \sum_{i \neq J} (B_{kJi} + B_{kiJ})_+ u_i + \sum_{i \neq J} (B_{kJi} + B_{kiJ})_- v_i + (B_{JJ})_+ u_J + (B_{JJ})_- v_J$$

$$\beta_1 = A_{kJ} + \sum_{i \neq J} (B_{kJi} + B_{kiJ})_+ v_i + \sum_{i \neq J} (B_{kJi} + B_{kiJ})_- u_i + (B_{JJ})_+ v_J + (B_{JJ})_- u_J$$

$$\begin{aligned}\gamma_0 = {} & lb_k - C_k - \sum_{i \neq J} (A_{ki})_+ v_i - \sum_{i \neq J} (A_{ki})_- u_i \\ & - \sum_{i \neq J, j \neq J, i \neq j} (B_{kji})_+ vxy(u_j, v_j, u_i, v_i) - \sum_{i \neq J, j \neq J, i \neq j} (B_{kji})_- \mu xy(u_j, v_j, u_i, v_i) \\ & - \sum_{i \neq J} (B_{kii})_+ vxx(u_i, v_i) - \sum_{i \neq J} (B_{kii})_- \mu xx(u_i, v_i)\end{aligned}$$

$$\begin{aligned}\gamma_1 = {} & ub_k - C_k - \sum_{i \neq J} (A_{ki})_+ u_i - \sum_{i \neq J} (A_{ki})_- v_i \\ & - \sum_{i \neq J, j \neq J, i \neq j} (B_{kji})_+ \mu xy(u_j, v_j, u_i, v_i) - \sum_{i \neq J, j \neq J, i \neq j} (B_{kji})_- vxy(u_j, v_j, u_i, v_i) \\ & - \sum_{i \neq J} (B_{kii})_+ \mu xx(u_i, v_i) - \sum_{i \neq J} (B_{kii})_- vxx(u_i, v_i)\end{aligned}$$

[0103] Form the set $\{\beta_0, \beta_1, \gamma_0, \gamma_1\}$. We now need to split up the set into the cases considered in Lemma B.4.

[0104] For each set $\{\beta_0, \beta_1, \gamma_0, \gamma_1\}$, if it is the case that $\beta_0 \leq 0 \leq \beta_1$, then split the set into two sets, $\{\beta_0, 0, \gamma_0, \gamma_1\}$ and $\{0, \beta_1, \gamma_0, \gamma_1\}$.

[0105] For each set $\{\beta_0, \beta_1, \gamma_0, \gamma_1\}$, if it is the case that $\gamma_0 \leq 0 \leq \gamma_1$, then split the set into two sets, $\{\beta_0, \beta_1, \gamma_0, 0\}$ and $\{\beta_0, \beta_1, 0, \gamma_1\}$.

[0106] Up to four separate sets may result from this procedure.

[0107] For each set $B^m = \{\beta_0, \beta_1, \gamma_0, \gamma_1\}$, use Lemma B.4 to compute a range $u'_m \le x_J \le v'_m$ for each set. Restrict each range by the original bounds (allowing for the possibility that the range will be empty).

$$u'_m = \max(u_J, u'_m)$$
$$v'_m = \min(v_J, v'_m)$$

[0108] Given the resulting sets of ranges $\{\{u'_m, v'_m\} : m = 1..M\}$, the variable will be in their union, $x_J \in \bigcup_m \{u'_m, v'_m\}$.

[0109] One can simplify them by sorting first by u'_m then by v'_m and then iteratively applying the following rule until no further simplifications are possible—

1) If $u'_m \le u'_{m+1}$ and $u'_{m+1} \le v'_m$ then

$\{\min(u'_m, u'_{m+1}), \max(v'_m, v'_{m+1})\} = \{u'_m, v'_m\} \cup \{u'_{m+1}, v'_{m+1}\}$ can replace both ranges.

2) If $v'_m < u'_m$, then delete the interval as being infeasible.

[0110] Given a set of multiple simplified ranges, there are two possible ways to apply them to the original node—

A) By conservatively bounding the multiple ranges, the node with bounds $N = \{\{u_J, v_J\}, \{u_i, v_i\} : i \ne J\}$ can be refined to

$N = \{\{\min_m(u'_m), \max_m(v'_m)\}; \{u_i, v_i\} : i \ne J\}$.

B) The node with bounds $N = \{\{u_J, v_J\}, \{u_i, v_i\} : i \ne J\}$ can be split into multiple refined nodes $N_m = \{\{u'_m, v'_m\}, \{u_i, v_i\} : i \ne J\}$.

C) A third option would be to look at the benefit of splitting according to (B) over using the simple bounds of (A) by computing the ratio of the ranges

$$\rho = \frac{\sum(v'_m - u'_m)}{\max_m(v'_m) - \min_m(u'_m)}.$$

and only splitting the node if the benefit is sufficient, say $\rho \le .5$ (the equivalent of one subdivision).

Non-Square-Free Bound Propagation

[0111] The procedure here is similar to that in the Square-Free Bound Propagation section, except that the squared term x_J^2 does appear in the function f_k, so the defined bounds $\beta_0 \le \beta \le \beta_1$ and $\gamma_0 \le \gamma \le \gamma_1$ will differ by a factor of the reciprocal of the coefficient of x_J^2, and Lemma B.5 will be invoked instead of Lemma B.4.

[0112] In order to apply the Lemmas, we first rearrange the inequalities into the form

$$lb_k - C_k \le A_{kJ}x_J + \left(\sum_{i\neq J}(B_{kJi} + B_{kiJ})x_i\right)x_J + B_{kJJ}x_J^2 + \sum_{i\neq J}A_{ki}x_i + \sum_{i\neq J, j\neq J}B_{kji}x_j x_i \le ub_k - C_k$$

and so if we define

$$\beta = \frac{\left(A_{kJ} + \sum_{i\neq J}(B_{kJi} + B_{kiJ})x_i\right)}{B_{kJJ}}$$

$$\gamma = \beta x_J + x_J^2$$

we also have

$$lb_k - C_k - \sum_{i\neq J}A_{ki}x_i - \sum_{i\neq J, j\neq J}B_{kji}x_j x_i \le B_{kJJ}\gamma \le ub_k - C_k - \sum_{i\neq J}A_{ki}x_i - \sum_{i\neq J, j\neq J}B_{kji}x_j x_i$$

[0113] We use the same bounding functions from the Square-Free Bound Propagation section.

[0114] We then define bounds $\beta_0 \le \beta \le \beta_1$ and $\gamma_0 \le \gamma \le \gamma_1$ for $\{\beta, \gamma\}$ based on the sign of B_{kJJ} —

$$\hat{\beta}_0 = A_{kJ} + \sum_{i\neq J}(B_{kJi} + B_{kiJ})_+ u_i + \sum_{i\neq J}(B_{kJi} + B_{kiJ})_- v_i$$

$$\hat{\beta}_1 = A_{kJ} + \sum_{i\neq J}(B_{kJi} + B_{kiJ})_+ v_i + \sum_{i\neq J}(B_{kJi} + B_{kiJ})_- u_i$$

$$\hat{\gamma}_0 = lb_k - C_k - \sum_{i\neq J}(A_{ki})_+ v_i - \sum_{i\neq J}(A_{ki})_- u_i$$
$$- \sum_{i\neq J, j\neq J, i\neq j}(B_{kji})_+ vxy(u_j, v_j, u_i, v_i) - \sum_{i\neq J, j\neq J, i\neq j}(B_{kji})_- \mu xy(u_j, v_j, u_i, v_i)$$
$$- \sum_{i\neq J}(B_{kii})_+ vxx(u_i, v_i) - \sum_{i\neq J}(B_{kii})_- \mu xx(u_i, v_i)$$

$$\hat{\gamma}_1 = ub_k - C_k - \sum_{i\neq J}(A_{ki})_+ u_i - \sum_{i\neq J}(A_{ki})_- v_i$$
$$- \sum_{i\neq J, j\neq J, i\neq j}(B_{kji})_+ \mu xy(u_j, v_j, u_i, v_i) - \sum_{i\neq J, j\neq J, i\neq j}(B_{kji})_- vxy(u_j, v_j, u_i, v_i)$$
$$- \sum_{i\neq J}(B_{kii})_+ \mu xx(u_i, v_i) - \sum_{i\neq J}(B_{kii})_- vxx(u_i, v_i)$$

If $B_{kJJ} > 0$

$$\beta_0 = \frac{\hat{\beta}_0}{B_{kJJ}}$$
$$\beta_1 = \frac{\hat{\beta}_1}{B_{kJJ}}$$
$$\gamma_0 = \frac{\hat{\gamma}_0}{B_{kJJ}}$$
$$\gamma_1 = \frac{\hat{\gamma}_1}{B_{kJJ}}$$

If $B_{kJJ} < 0$

$$\beta_0 = \frac{\hat{\beta}_1}{B_{kJJ}}$$
$$\beta_1 = \frac{\hat{\beta}_0}{B_{kJJ}}$$
$$\gamma_0 = \frac{\hat{\gamma}_1}{B_{kJJ}}$$
$$\gamma_1 = \frac{\hat{\gamma}_0}{B_{kJJ}}$$

[0115] We now follow the same procedure as in the Square-Free Bound Propagation section, except that we apply Lemma B.5 to the generation of the ranges $x_J \in \bigcup_m \{u'_m, v'_m\}$. Note that unlike Lemma B.4, each case in Lemma B.5

may themselves generate multiple or empty ranges.

Proofs for Nonlinear Function Problems

[0116] For the time being, let us consider only feasibility, although the same approach applies to optimization as well.

[0117] Let $f(x):\Re^n \to \Re^m$ be a function with Lipschitz-continuous derivatives.

[0118] Assume we have a trial solution $\bar{x}$. By the mean value theorem, for every point x there exists a point $\tilde{x}$ (on the segment between $\bar{x}$ and x) such that

$$f_k(x) = f_k(\bar{x}) + \nabla f_k(\tilde{x})(x - \bar{x}) \tag{A.1}$$

[0119] Within a set of point bounds,

$$u \le x \le v \tag{A.2}$$

there exist gradient bounds

$$\begin{aligned} F_k(u,v) &= \inf\{\nabla f_k(x) : u \le x \le v\} \\ G_k(u,v) &= \sup\{\nabla f_k(x) : u \le x \le v\} \end{aligned} \tag{A.3}$$

so that

$$u \le x \le v \;\Rightarrow\; F \le \nabla f(x) \le G \tag{A.4}$$

(For notational convenience, we will suppress the functional dependence of F and G on u and v.)

[0120] We will assume that the trial solution satisfies the bounds

$$u \le \bar{x} \le v \tag{A.5}$$

[0121] If we define

$$\begin{aligned} \bar{u} &= u - \bar{x} \\ \bar{v} &= v - \bar{x} \\ \bar{F} &= F - \nabla f(\bar{x}) \\ \bar{G} &= G - \nabla f(\bar{x}) \end{aligned} \tag{A.6}$$

we will get the enveloping inequalities

$$\bar{u} \le x - \bar{x} \le \bar{v} \;\Rightarrow \tag{A.7}$$

$$f(\bar{x}) + \nabla f(\bar{x})(x-\bar{x}) + F(x-\bar{x})_+ + G(x-\bar{x})_-$$
$$\leq f(x) \leq$$
$$f(\bar{x}) + \nabla f(\bar{x})(x-\bar{x}) + G(x-\bar{x})_+ + F(x-\bar{x})_-$$

[0122] Now consider the following series of problems—

$$\text{(A.8)} \quad P0 = \left\{ x: \begin{array}{l} u^0 \leq x \leq v^0 \\ lb \leq f(x) \leq ub \end{array} \right\}$$

$$\text{(A.9)} \quad Pbd(u,v) = \left\{ x: \begin{array}{l} u \leq x \leq v \\ lb \leq f(x) \leq ub \end{array} \right\}$$

$$\text{(A.10)} \; LP(\bar{x},u,v) = \left\{ x,z,w: \begin{array}{l} \bar{u} \leq x-\bar{x} \leq \bar{v} \\ lb \leq f(\bar{x}) + \nabla f(\bar{x})(x-\bar{x}) + \bar{G}z - \bar{F}w \\ ub \geq f(\bar{x}) + \nabla f(\bar{x})(x-\bar{x}) + \bar{F}z - \bar{G}w \\ x-\bar{x} = z-w \\ z+w \leq \max(|\bar{v}|,|\bar{u}|) \\ z,w \geq 0 \end{array} \right\}$$

[0123] Define the *infeasibility* of a particular point to be

$$\text{(A.11)} \; \Delta_k(x) = \max\left((f_k(x) - ub_k)_+, (lb_k - f_k(x))_+\right)$$

(As a result $\Delta(x) \geq 0$, and $\Delta(x) = 0$ only if the point x is feasible.)

[0124] Theorem 1—

T1-1) For every $x \in P0$, there exist bounds $\{u,v\}$ such that

$x \in Pbd(u,v)$.

T1-2) $Pbd(u,v) \subseteq LP(\bar{x},u,v)$.

T1-3) For every $\{x,u,v\} \in LP(\bar{x},u,v)$,

$$\Delta(x) \leq (G-F)(z+w) \leq (G-F)\max(|\bar{u}|,|\bar{v}|) \leq (G-F)(v-u).$$

[0125] Corollary 1—

$$Pbd(u,v) \cap LP(\bar{x},u,v) = Pbd(u,v) \cap LP(x',u,v),$$

[0126] For the quadratic problem,

$$\text{(A.12)} \; f(x) = C + Ax + Bxx$$

and we can find

$$\tilde{x} = \tfrac{1}{2}(x + \bar{x})$$

$$\text{(A.13)}\quad \nabla f(\bar{x}) = \bar{A} = A + (B + B^*)\bar{x}$$

$$\nabla f(\tilde{x}) - \nabla f(\bar{x}) = B(x - \bar{x})$$

so we can compute a set of gradient bounds

$$\text{(A.14)}\quad \begin{aligned} F &= (B)_+ u + (B)_- v \\ G &= (B)_+ v + (B)_- u \end{aligned}$$

so that

$$\text{(A.15)}\quad u \le x \le v \;\Rightarrow\; F \le Bx \le G \;\Rightarrow$$

$$\bar{F} = F - B\bar{x} \;\le\; B(x - \bar{x}) = \nabla f(\tilde{x}) - \nabla f(\bar{x}) \;\le\; G - B\bar{x} = \bar{G}\,.$$

Bounding Lemmas

[0127] Lemma B.1—

Let $z = xy$. If $x_0 \le x \le x_1$ and $y_0 \le y \le y_1$, then

$$\min(x_0y_0, x_1y_0, x_0y_1, x_1y_1) \le z \le \max(x_0y_0, x_1y_0, x_0y_1, x_1y_1)$$

[0128] Lemma B.2—

Let $z = x^2$. If $x_0 \le x \le x_1$, then

$$\min({x_0}^2, (x_0x_1)_+, {x_1}^2) \le z \le \max({x_0}^2, (x_0x_1)_+, {x_1}^2)$$

[0129] Or equivalently, Lemma B.3—

Let $z = x^2$. If $x_0 \le x \le x_1$, then

a) If $x_0 \le x_1 \le 0$ or $0 \le x_0 \le x_1$ then

$$\min({x_0}^2, {x_1}^2) \le z \le \max({x_0}^2, {x_1}^2)$$

b) If $x_0 \le 0 \le x_1$ then

$$0 \le z \le \max({x_0}^2, {x_1}^2)$$

[0130] Lemma B.4—

Let $\gamma = \beta x$. If $\beta_0 \le \beta \le \beta_1$ and $\gamma_0 \le \gamma \le \gamma_1$, then

I) If $0 \le \beta_0 \le \beta_1$ and $0 \le \gamma_0 \le \gamma_1$ then

$$\frac{\gamma_0}{\beta_1} \le x \le \frac{\gamma_1}{\beta_0} \qquad (\frac{\gamma_0}{\beta_1} \le x \le \infty \text{ if } \beta_0 = 0)$$

II) If $\beta_0 \le \beta_1 \le 0$ and $0 \le \gamma_0 \le \gamma_1$, then

$$\frac{\gamma_1}{\beta_1} \le x \le \frac{\gamma_0}{\beta_0} \qquad (-\infty \le x \le \frac{\gamma_0}{\beta_0} \text{ if } \beta_1 = 0)$$

III) If $0 \le \beta_0 \le \beta_1$ and $\gamma_0 \le \gamma_1 \le 0$ then

$$\frac{\gamma_0}{\beta_0} \le x \le \frac{\gamma_1}{\beta_1} \qquad (-\infty \le x \le \frac{\gamma_1}{\beta_1} \text{ if } \beta_0 = 0)$$

IV) If $\beta_0 \le \beta_1 \le 0$ and $\gamma_0 \le \gamma_1 \le 0$, then

$$\frac{\gamma_1}{\beta_0} \le x \le \frac{\gamma_0}{\beta_1} \qquad (\frac{\gamma_1}{\beta_0} \le x \le \infty \text{ if } \beta_1 = 0)$$

[0131] The following is an equivalent form, more convenient in some cases

Lemma B.4'—

Let $\gamma = \beta x$. If $\beta_0 \le \beta \le \beta_1$ and $\gamma_0 \le \gamma \le \gamma_1$, then

I,III) If $0 \le \beta_0 \le \beta_1$ then

$$\min\left(\frac{\gamma_0}{\beta_0}, \frac{\gamma_0}{\beta 1}\right) \le x \le \max\left(\frac{\gamma_1}{\beta_0}, \frac{\gamma_1}{\beta_1}\right) \qquad (\frac{1}{\beta_0} \approx \infty \text{ if } \beta_0 = 0)$$

II,IV) If $\beta_0 \le \beta_1 \le 0$, then

$$\min\left(\frac{\gamma_1}{\beta_0}, \frac{\gamma_1}{\beta_1}\right) \le x \le \max\left(\frac{\gamma_0}{\beta_0}, \frac{\gamma_0}{\beta_1}\right) \qquad (\frac{1}{\beta_1} \approx -\infty \text{ if } \beta_1 = 0)$$

[0132] Lemma B.5— (Note that Cases I and II result in identical results.)

Let $\gamma = \beta x + x^2$. If $\beta_0 \le \beta \le \beta_1$ and $\gamma_0 \le \gamma \le \gamma_1$, then

I) If $0 \le \beta_0 \le \beta_1$ and $0 \le \gamma_0 \le \gamma_1$ then

$$-\frac{\beta_1}{2} - \sqrt{\frac{\beta_1^2}{4} + \gamma_1} \le x \le -\frac{\beta_0}{2} + \sqrt{\frac{\beta_0^2}{4} + \gamma_1}$$

or more accurately,

$$\text{either } -\frac{\beta_1}{2} - \sqrt{\frac{\beta_1^2}{4} + \gamma_1} \le x \le -\frac{\beta_0}{2} - \sqrt{\frac{\beta_0^2}{4} + \gamma_0}$$

or $-\frac{\beta_1}{2}+\sqrt{\frac{\beta_1^2}{4}+\gamma_0} \le x \le -\frac{\beta_0}{2}+\sqrt{\frac{\beta_0^2}{4}+\gamma_1}$

II) If $\beta_0 \le \beta_1 \le 0$ and $0 \le \gamma_0 \le \gamma_1$, then

$$-\frac{\beta_1}{2}-\sqrt{\frac{\beta_1^2}{4}+\gamma_1} \le x \le -\frac{\beta_0}{2}+\sqrt{\frac{\beta_0^2}{4}+\gamma_1}$$

or more accurately,

either $-\frac{\beta_1}{2}-\sqrt{\frac{\beta_1^2}{4}+\gamma_1} \le x \le -\frac{\beta_0}{2}-\sqrt{\frac{\beta_0^2}{4}+\gamma_0}$

or $-\frac{\beta_1}{2}+\sqrt{\frac{\beta_1^2}{4}+\gamma_0} \le x \le -\frac{\beta_0}{2}+\sqrt{\frac{\beta_0^2}{4}+\gamma_1}$

III) If $0 \le \beta_0 \le \beta_1$ and $\gamma_0 \le \gamma_1 \le 0$ then

III.a) If $\frac{\beta_1^2}{4}+\gamma_1 < 0$, then there are no possible solutions for x.

III.b) If $\frac{\beta_1^2}{4}+\gamma_1 \ge 0$, then

$$-\frac{\beta_1}{2}-\sqrt{\frac{\beta_1^2}{4}+\gamma_1} \le x \le -\frac{\beta_1}{2}+\sqrt{\frac{\beta_1^2}{4}+\gamma_1}$$

or more accurately,

either $-\frac{\beta_1}{2}-\sqrt{\frac{\beta_1^2}{4}+\gamma_1} \le x \le -\frac{\beta_0}{2}-\sqrt{\left(\frac{\beta_0^2}{4}+\gamma_0\right)_+}$

or $-\frac{\beta_0}{2}+\sqrt{\left(\frac{\beta_0^2}{4}+\gamma_0\right)_+} \le x \le -\frac{\beta_1}{2}+\sqrt{\frac{\beta_1^2}{4}+\gamma_1}$

IV) If $\beta_0 \le \beta_1 \le 0$ and $\gamma_0 \le \gamma_1 \le 0$, then

III.a) If $\frac{\beta_0^2}{4}+\gamma_1 < 0$, then there are no possible solutions for x.

III.b) If $\frac{\beta_0^2}{4}+\gamma_1 \ge 0$, then

$$-\frac{\beta_0}{2}-\sqrt{\frac{\beta_0^{\,2}}{4}+\gamma_1}\le x\le -\frac{\beta_0}{2}+\sqrt{\frac{\beta_0^{\,2}}{4}+\gamma_1}$$

or more accurately,

$$\text{either } -\frac{\beta_0}{2}-\sqrt{\frac{\beta_0^{\,2}}{4}+\gamma_1}\le x\le -\frac{\beta_1}{2}-\sqrt{\left(\frac{\beta_1^{\,2}}{4}+\gamma_0\right)_+}$$

$$\text{or } -\frac{\beta_1}{2}+\sqrt{\left(\frac{\beta_1^{\,2}}{4}+\gamma_0\right)_+}\le x\le -\frac{\beta_0}{2}+\sqrt{\frac{\beta_0^{\,2}}{4}+\gamma_1}$$

Gradient Bound Subdivision

[0133] It is also viable for the quadratic-only problem to subdivide the gradient bounds directly, e.g. by choosing a dimension $\{k',i'\}$ and subdividing that gradient bound, e.g. at $H_{ki}=\frac{F_{ki}+G_{ki}}{2}$, or $H_{ki}=(B\bar{x})_{ki}$ if subdivision at the linearization points is desired.. In that case, (1.9) no longer applies, so it is necessary to include the gradient inequalities $F\le Bx\le G$ directly in the problems (1.23) - (1.28) as explicit inequalities. Theorem 1 and Corollary 1 will still apply.

[0134] If one is subdividing the gradients, then there is additional bound propagation between the point bounds and the gradient bounds that can be defined—

1) To refine the gradient bounds

$$F'=\max\left(F,(B)_+u+(B)_-v\right)$$

$$G'=\min\left(G,(B)_+v+(B)_-u\right)$$

2) To refine the point upper bounds

$$\forall j : v'_j = v_j$$

$$\forall k,i,j : \begin{cases} B_{kji} > 0 \Rightarrow v'_j = \min\left(v'_j, \dfrac{G'_{ki} - \sum_{\hat{j}\neq j} (B_{k\hat{j}i})_+ u'_{\hat{j}} - \sum_{\hat{j}\neq j} (B_{k\hat{j}i})_- v'_{\hat{j}}}{B_{kji}} \right) \\ B_{kji} < 0 \Rightarrow v'_j = \min\left(v'_j, \dfrac{F'_{ki} - \sum_{\hat{j}\neq j} (B_{k\hat{j}i})_+ v'_{\hat{j}} - \sum_{\hat{j}\neq j} (B_{k\hat{j}i})_- u'_{\hat{j}}}{B_{kji}} \right) \end{cases}$$

To refine the point lower bounds

$$\forall j : u'_j = u_j$$

$$\forall k,i,j : \begin{cases} B_{kji} > 0 \Rightarrow u'_j = \max\left(u'_j, \dfrac{F'_{ki} - \sum_{\hat{j}\neq j} (B_{k\hat{j}i})_+ v'_{\hat{j}} - \sum_{\hat{j}\neq j} (B_{k\hat{j}i})_- u'_{\hat{j}}}{B_{kji}} \right) \\ B_{kji} < 0 \Rightarrow u'_j = \max\left(u'_j, \dfrac{G'_{ki} - \sum_{\hat{j}\neq j} (B_{k\hat{j}i})_+ u'_{\hat{j}} - \sum_{\hat{j}\neq j} (B_{k\hat{j}i})_- v'_{\hat{j}}}{B_{kji}} \right) \end{cases}$$

Robust Linear Propagation

[0135] The following modifies (2) in the Simple Propagation of Bounds to account for possible numerical problems. Let $\varepsilon > 0$ be our numerical tolerance. For every linear constraint

$$k : f_k(x) = C_k + \sum_i A_{ki} x_i$$

and for every variable x_i which appears in f_k and such that $|A_{ki}| > \varepsilon$, we define the following at each stage of the iteration where the bounds at that iteration are $\{u, v\}$.

$$p_{ki} = lb_k - C_k - \sum_{j\neq i} (A_{kj})_+ v_j - \sum_{j\neq i} (A_{kj})_- u_j$$

$$q_{ki} = ub_k - C_k - \sum_{j\neq i} (A_{kj})_+ u_j - \sum_{j\neq i} (A_{kj})_- v_j$$

$$\Delta_{ki} = \varepsilon\left(1 + \sum_{j\neq i} \left(|A_{kj}| + \max\left(|u_j|, |v_j| \right) \right) \right)$$

We can determine then determine new point bounds

$$\forall k \in K_1, f_k \text{ linear}, \forall i: \begin{cases} A_{ki} > \varepsilon \Rightarrow v_i' = \max\left(u_j - |1+u_j|, \min\left(v_i, \max\left(\frac{q_{ki}+\Delta_{ki}}{A_{ki}-\varepsilon}, \frac{q_{ki}+\Delta_{ki}}{A_{ki}+\varepsilon}\right)\right)\right) \\ A_{ki} < -\varepsilon \Rightarrow v_i' = \max\left(u_j - |1+u_j|, \min\left(v_i, \max\left(\frac{p_{ki}+\Delta_{ki}}{A_{ki}-\varepsilon}, \frac{p_{ki}+\Delta_{ki}}{A_{ki}+\varepsilon}\right)\right)\right) \end{cases}$$

$$\forall k \in K_1, f_k \text{ linear}, \forall i: \begin{cases} A_{ki} > \varepsilon \Rightarrow u_i' = \min\left(v_j + |1+v_j|, \max\left(u_i, \min\left(\frac{p_{ki}+\Delta_{ki}}{A_{ki}-\varepsilon}, \frac{p_{ki}+\Delta_{ki}}{A_{ki}+\varepsilon}\right)\right)\right) \\ A_{ki} < -\varepsilon \Rightarrow u_i' = \min\left(v_j + |1+v_j|, \max\left(u_i, \min\left(\frac{q_{ki}+\Delta_{ki}}{A_{ki}-\varepsilon}, \frac{q_{ki}+\Delta_{ki}}{A_{ki}+\varepsilon}\right)\right)\right) \end{cases}$$

Numerically Robust Square-Free Bound Propagation

[0136] The following modifies the Square-Free Bound Propagation to account for possible numerical problems. Let $\varepsilon > 0$ be our numerical tolerance.

[0137] For convenience, define the following error bound on the product xy

$$\Delta xy(x_0, x_1, y_0, y_1) = \varepsilon \max(|x_0|, |y_0|, |x_1|, |y_1|)$$

[0138] From the Bounding Lemmas, we define the robust bounding functions for multiply and square—

$$\mu^{\varepsilon} xy(x_0, x_1, y_0, y_1) = \min(x_0 y_0, x_1 y_0, x_0 y_1, x_1 y_1) - \Delta xy(x_0, x_1, y_0, y_1)$$

$$v^{\varepsilon} xy(x_0, x_1, y_0, y_1) = \max(x_0 y_0, x_1 y_0, x_0 y_1, x_1 y_1) + \Delta xy(x_0, x_1, y_0, y_1)$$

$$\mu^{\varepsilon} xx(x_0, x_1) = \min({x_0}^2, (x_0 x_1)_+, {x_1}^2) - \Delta xy(x_0, x_1, x_0, x_1)$$

$$v^{\varepsilon} xx(x_0, x_1) = \max({x_0}^2, (x_0 x_1)_+, {x_1}^2) + \Delta xy(x_0, x_1, x_0, x_1)$$

We then define rigorous bounds $\beta_0^{\varepsilon} \le \beta \le \beta_1^{\varepsilon}$ and $\gamma_0^{\varepsilon} \le \gamma \le \gamma_1^{\varepsilon}$ for $\{\beta, \gamma\}$—

$$\beta_0^{\varepsilon} = A_{kJ} - \varepsilon + \sum_{i \ne J} (B_{kJi} + B_{kiJ} - \varepsilon)_+ (u_i - \varepsilon) + \sum_{i \ne J} (B_{kJi} + B_{kiJ} - \varepsilon)_- (v_i + \varepsilon) + (B_{JJ} - \varepsilon)_+ (u_J - \varepsilon) -$$

$$\beta_1^{\varepsilon} = A_{kJ} + \varepsilon + \sum_{i \ne J} (B_{kJi} + B_{kJi} + \varepsilon)_+ (v_i + \varepsilon) + \sum_{i \ne J} (B_{kJi} + B_{kiJ} + \varepsilon)_- (u_i - \varepsilon) + (B_{JJ} + \varepsilon)_+ (v_J + \varepsilon) -$$

$$\gamma_0^\varepsilon = lb_k - C_k - \varepsilon - \sum_{i\neq J}(A_{ki}+\varepsilon)_+(v_i+\varepsilon) - \sum_{i\neq J}(A_{ki}+\varepsilon)_-(u_i-\varepsilon)$$
$$- \sum_{i\neq J, j\neq J, i\neq j}(B_{kji}+\varepsilon)_+ v^\varepsilon xy(u_j,v_j,u_i,v_i) - \sum_{i\neq J, j\neq J, i\neq j}(B_{kji}+\varepsilon)_- \mu^\varepsilon xy(u_j,v_j,u_i,v_i)$$
$$- \sum_{i\neq J}(B_{kai}+\varepsilon)_+ v^\varepsilon xx(u_i,v_i) - \sum_{i\neq J}(B_{kii}+\varepsilon)_- \mu^\varepsilon xx(u_i,v_i)$$

$$\gamma_1^\varepsilon = ub_k - C_k + \varepsilon - \sum_{i\neq J}(A_{ki}-\varepsilon)_+(u_i-\varepsilon) - \sum_{i\neq J}(A_{ki}-\varepsilon)_-(v_i+\varepsilon)$$
$$- \sum_{i\neq J, j\neq J, i\neq j}(B_{kji}-\varepsilon)_+ \mu^\varepsilon xy(u_j,v_j,u_i,v_i) - \sum_{i\neq J, j\neq J, i\neq j}(B_{kji}-\varepsilon)_- v^\varepsilon xy(u_j,v_j,u_i,v_i)$$
$$- \sum_{i\neq J}(B_{kii}-\varepsilon)_+ \mu^\varepsilon xx(u_i,v_i) - \sum_{i\neq J}(B_{kii}-\varepsilon)_- v^\varepsilon xx(u_i,v_i)$$

We now follow the identical process as in (2.8) replacing the original set $\{\beta_0,\beta_1,\gamma_0,\gamma_1\}$ by the set $\{\beta_0^\varepsilon,\beta_1^\varepsilon,\gamma_0^\varepsilon,\gamma_1^\varepsilon\}$.

Numerically Robust Non-Square-Free Bound Propagation

[0139] The following modifies the Non-Square-Free Bound Propagation to account for possible numerical problems. Let $\varepsilon > 0$ be our numerical tolerance. Define the robust bounding functions for multiply and square as in the Numerically Robust Square-Free Bound Propagation section.

[0140] We define rigorous bounds $\beta_0^\varepsilon \le \beta \le \beta_1^\varepsilon$ and $\gamma_0^\varepsilon \le \gamma \le \gamma_1^\varepsilon$ for $\{\beta,\gamma\}$ based on the sign of B_{kJJ} —

$$\hat{\beta}_0^\varepsilon = A_{kJ} - \varepsilon + \sum_{i\neq J}(B_{kJi}+B_{kiJ}-\varepsilon)_+(u_i-\varepsilon) + \sum_{i\neq J}(B_{kJi}+B_{kiJ}-\varepsilon)_-(v_i+\varepsilon)$$
$$\hat{\beta}_1^\varepsilon = A_{kJ} + \varepsilon + \sum_{i\neq J}(B_{kJi}+B_{kiJ}+\varepsilon)_+(v_i+\varepsilon) + \sum_{i\neq J}(B_{kJi}+B_{kiJ}+\varepsilon)_-(u_i-\varepsilon)$$

$$\hat{\gamma}_0^\varepsilon = lb_k - C_k - \varepsilon - \sum_{i\neq J}(A_{ki}+\varepsilon)_+(v_i+\varepsilon) - \sum_{i\neq J}(A_{ki}+\varepsilon)_-(u_i-\varepsilon)$$
$$- \sum_{i\neq J, j\neq J, i\neq j}(B_{kji}+\varepsilon)_+ v^\varepsilon xy(u_j,v_j,u_i,v_i) - \sum_{i\neq J, j\neq J, i\neq j}(B_{kji}+\varepsilon)_- \mu^\varepsilon xy(u_j,v_j,u_i,v_i)$$
$$- \sum_{i\neq J}(B_{kii}+\varepsilon)_+ v^\varepsilon xx(u_i,v_i) - \sum_{i\neq J}(B_{kii}+\varepsilon)_- \mu^\varepsilon xx(u_i,v_i)$$

$$\hat{\gamma}_1^{\varepsilon} = ub_k - C_k + \varepsilon - \sum_{i \neq J} (A_{ki} - \varepsilon)_+ (u_i - \varepsilon) - \sum_{i \neq J} (A_{ki} - \varepsilon)_- (v_i + \varepsilon)$$
$$- \sum_{i \neq J, j \neq J, i \neq j} (B_{kji} - \varepsilon)_+ \mu^{\varepsilon} xy(u_j, v_j, u_i, v_i) - \sum_{i \neq J, j \neq J, i \neq j} (B_{kji} - \varepsilon)_- \nu^{\varepsilon} xy(u_j, v_j, u_i, v_i)$$
$$- \sum_{i \neq J} (B_{kii} - \varepsilon)_+ \mu^{\varepsilon} xx(u_i, v_i) - \sum_{i \neq J} (B_{kii} - \varepsilon)_- \nu^{\varepsilon} xx(u_i, v_i)$$

[0141] We now follow the same process as in the Non-Square-Free Bound Propagation section replacing the original set $\{\hat{\beta}_0, \hat{\beta}_1, \hat{\gamma}_0, \hat{\gamma}_1\}$ by the set $\{\hat{\beta}_0^{\varepsilon}, \hat{\beta}_1^{\varepsilon}, \hat{\gamma}_0^{\varepsilon}, \hat{\gamma}_1^{\varepsilon}\}$.

Bound Propagation for Mixing Functions

[0142] A common equation for refinery and chemical processes is a mixing equation of the form

(G.1) $x_0(y_1 + y_2 + \ldots + y_n) = x_1 y_1 + x_2 y_2 + \ldots + x_n y_n$

where

$y_1 \geq 0, y_2 \geq 0, \ldots, y_n \geq 0$.

[0143] Here we are given materials indexed by $\{1,2,\ldots,n\}$, and recipe sizes (or rates, or masses, or ratios, whatever mixing unit of interest) $\{y_1, y_2, \ldots, y_n\}$ for a mix of the materials. Then (G.1) represents a simple mixing model for a property x_0 of the resulting mix given values $\{x_1, x_2, \ldots, x_n\}$ for the property of each of the materials.

[0144] In this case, the resulting property will be a convex linear combination of the individual properties. Hence if we are given bounds

(G.2) $a_1 \leq x_1 \leq b_1, a_2 \leq x_2 \leq b_2, \ldots, a_n \leq x_n \leq b_n$

we can derive the bounds

(G.3) $$\begin{aligned} x_0 &\geq \min(a_1, a_2, \ldots, a_n) \\ x_0 &\leq \max(b_1, b_2, \ldots, b_n) \end{aligned}.$$

[0145] It is to be understood that the above description it is intended to be illustrative, and not restrictive. Many other embodiments are possible and some

will be apparent to those skilled in the art, upon reviewing the above description. For example the local linear bounding and the local linearization can be performed in the opposite order, and more. Therefore, the spirit and scope of the appended claims should not be limited to the above description. The scope of the invention should be determined with reference to the appended claims, along with the full scope of equivalents to which such claims are entitled.